谷歌Android作为DTV解码器系统的软件环境

Nikola Kuzmanovic

Faculty of Technical Sciences

Novi Sad, Serbia

[nikola.kuzmanovic@rt-rk.com](mailto:nikola.kuzmanovic@rt-rk.com)

Tomislav Maruna, Milan Savic, Goran Miljkovic,

Djordje Isailovic

RT-RK d.o.o.

Novi Sad, Serbia

#### 摘要*—*文提出了一种使用谷歌Android软件栈作为DTV(电视集)和机顶盒应用环境的方法。通过扩展Android软件栈来支持DTV解码器设备，同时为Android环境开发万能电视和机顶盒应用，由此可加速软件的发展并且缩短下一代产品的发布周期。本文介绍了将完整的Android软件栈（包括Android linux内核，系统驱动程序，用户空间库，Android特定的库和程序）移植到数字电视解码器系统上（主要是利用Linux内核和稳定的设备驱动程序）的过程，实现的扩展将支持数字电视频道搜索，记忆，播放和录音。最终的系统性能和特点由本文展现给我们。

**关键字**—数字电视机，机顶盒，Linux内核，嵌入式系统，Android软件栈

#### 简介

今天，DTV（数字电视）和机顶盒（进一步说是文本多媒体设备）已经由自定义的专用软件（操作系统和应用程序）来控制，一般说来，是一个特定的模式或者是一个家庭的同类模式。不同的多媒体设备模型之间的代码可重用性是很小的，因此一个多媒体设备模型的开发周期，需要大量的时间和资源。

手持设备和现代化的家庭多媒体设备之间的差异不断减少，同时，使用的硬件相似性正在变大（如CPU的处理能力，RAM大小，闪存大小以及其他集成设备的可用性如：LAN和Wi-Fi无线局域网，蓝牙，硬件加速图形等）。虽然更新、更先进的功能不断地加入到它们中（如视频录制，网络电视，网上冲浪，等等），但在这一段时间内，手持设备和多媒体设备的主要目的并没有改变。

Android是第一个为移动设备开发而建立的软件平台，它是完全开放和免费的，由开放手机联盟（以谷歌为首的超过30个公司的集团）发展起来。Android在嵌入式市场变得越来越流行，主要原因有两个。首先，源代码是完全免费的；而且使用Java虚拟机没有特权使用费。第二，源于第一，Android非常适合扩展和增强。

本文提出了一种使用谷歌Android软件栈作为嵌入式多媒体设备（主要是数字电视解码器系统和机顶盒）的应用环境的方法。进行Android的移植和修改的动力在于它的可用性（开放源码）以及它使用的是Linux内核的硬件抽象层（开放源码）。Android软件栈和Linux内核，使开发人员能够访问多媒体，网络连接，图形，电话等，这使得在开发不同的Android设备上普遍使用的应用程序的过程中，不需要考虑特定的硬件系统上应用程序是否可以运行的问题。

#### 安卓软件栈



图1安卓系统架构

Android的系统核心服务依赖于Linux内核，如安全性，内存管理，进程管理，网络协议栈和驱动模型。同时Linux内核也在硬件和剩余的软件栈之间扮演抽象层的角色。

Android的原生库是用C / C ++开发的，并且被Android系统的各个组成部分所使用，通过Android应用程序框架将它的特性暴露给开发者。一些核心库如：系统C库（一个针对基于嵌入式Linux设备，用BSD派生实现的标准C系统库（libc）），媒体库（基于PureVideo OPENCORE，主要针对流行音频和视频的播放和录制，以及图像文件的处理），WebKit库（一个众所周知的现代Web浏览器引擎，现在作为Android浏览器引擎），OpenGL ES 3D的库（这个库是使用硬件3D加速或者高度优化的3D软件光栅来实现的），FreeType字体（位图和矢量字体渲染），SQLite（一个强大的和轻量级的关系数据库引擎，用于所有应用程序）等。

Android运行时包括了一套核心库，这套核心库提供了大量Java核心库所具有的功能，Android应用程序利用Dalvik虚拟机（egisterbased VM，运行用Java编译器编译的类，这些类被转化为Dalvik可执行（.dex）的格式，并针对嵌入式设备进行了优化（占用很小的内存空间））运行在单独的进程上。Dalvik虚拟机基本功能的实现利用的是Linux内核，如线程和低级别的内存管理。

Android使开发者能够完全访问核心应用程序所使用的框架API，开发者可以自由免费地使用硬件设备、地理位置信息、运行后台服务、设置闹钟、添加通知、重用和替换现有的组件。简单的说，Android提供了一个完全开放的免费的开发平台，使开发人员能够开发出先进的、功能强大的应用程序。

Android的最高层是应用层，这一层可容纳许多内置的应用程序如：SMS程序，电子邮件客户端，日历，地图，浏览器，电话等，另外还包括从网上下载或通过SD卡安装的特定的应用程序。

#### 系统概述

这个项目的目的就是将Android系统移植到一个真正的数字电视解码器（Micronas公司的IDTV开发平台）设备上，这个设备包括一个集成的系统芯片（SoC）和外围设备，其中SoC包括了MIPS24kc（240MHz的Linux专用处理器），解码处理器（多路A/ V专用），图像处理单元（GPU），图像加速器（GA），通用串行总线（USB）控制器等。可用的软件有： 2.6.21.7 版本的Linux内核（具有稳定的驱动程序，适用于在SoC上的所有设备），GNU的gcc/ glibc交叉编译工具，支持DirectFB的图形库等等。

#### Android****移植****

移植Android系统到大端MIPS平台上需要三个步骤，利用的是Micronas平台和2.6.21.7版本的（这是一个在今天看起来很老的版本）Linux内核。尽管如此，平台驱动程序所使用的框架，使得将它们移植到Android2.6.27或者更高的内核版本上（相对于它的逆过程来说）变得更难（工作量超过了这个项目的预期），因此，在第一个阶段，Android软件栈所使用的Linux内核驱动程序需要移植到该平台所提供的内核上，第二个阶段，则需要扩展Android编译环境以支持大端MIPS平台，在编译过程中，从环境得到的配置必须正确地编译。最后阶段，则包括音频驱动的集成和DirectFB视频帧缓冲的支持。

1. **Linux内核移植**

为了支持Android软件栈，Linux内核树需要集成一些新的驱动程序，这些驱动程序有：

* ASHMEM（匿名共享内存）——共享内存管理器，类似于POSIX SHM，但有不同的行为，它支持简单的基于文件映射的API。
* BINDER ——类似CORBA的IPC通信机制。
* Android event logger——Linux内核驱动程序，直接在内核中记录调试信息。

为支持Android软件栈，2.6.21.7版本的Linux内核树配置参数以及所使用的平台，需要在许多方面进行修改，并且，上述提及的新的驱动程序也需要集成。

1. **Android 环境修改**

Android系统已经成功移植到了小端MIPS平台上，但是由于所使用的平台（大端平台）与小端平台的不同，对Android系统的修改还有很多。

* 第一种修改涉及平台本地的一些二进制文件，主要是开源库和用C / C ++编写的程序（endianes支持通常是通过内部编译系统（主要是“autoconf ”）实现的），同时修改Android编译系统的makefiles 文件（Android.mk）来支持大端变量和定义。
* 第二种修改涉及编译工具的代码尾数不能识别的问题，这造成的错误是罕见的，并且很难发现，它们通常会导致Android编译失败或者平台重启失败，受影响的程序有Android pre-linker Apriori、stripping application Soslim、resource manager aapt（Android资源打包工具）等。
* 第三种也是最后一种修改，包括添加一些在Android上运行的程序和支持代码尾数识别的库，例如Dalvik VM（即明确支持大端的平台）使用的一些数据类型的指针分配在小端体系结构中工作正常，但是很难移植到大端平台上，罪魁祸首是使用贯穿整个VM代码的JValue结构作为许多值类型容器的结构，这是典型的误用小端系统中数据类型的指针分配，在日志文件中找也到了相同类型的错误。其他的一些修改包括颜色问题的矫正，尾数排序，SKIA引擎，像素和Surface Flinger。

1. **Android DirectFB支持**

DirectFB（直接帧缓存）框架是一个基于GNU / Linux/ UNIX、在Linux的帧缓冲设备之上的图形软件库（占用很小的内存空间），它提供了图形加速和输入设备处理等功能。由于它是开源的，所以很多DTV SoC厂商用它来实现自己的帧缓冲IP-S驱动程序。

Android支持用标准的Linux帧缓冲设备驱动程序作为主要的绘图表面。随着DTV SoC供应商实现硬件加速的DirectFB视频驱动程序，Android系统对DirectFB的支持已经遍及系统不同层次。首先，支持UI的绘制，其次，通过图形硬件加速来绘制图形。

* DirectFB图形分配设备实现了支持主视频帧缓冲内存的分配，以及所有其他种类的多颜色空间的表面分配。
* DirectFB的copybit库还实现了支持硬件加速的2D位块传送操作。如果提供的DirectFB驱动程序支持2D位块传输操作（不同颜色空间之间的blit操作和stretch-blit操作），那么中央CPU单元将会从耗时的块传输任务过程中解放出来。

使用DFB作为UI 图形缓冲的唯一缺点是它本身不支持3D加速（Android是可以使用的）。但是过去几年稳步发展的DirectFB图形库（有尝试过添加像3D加速一样的OpenGL）使我们相信如果硬件3D加速的IP出现的话，基于DirectFB图形库的3D硬件加速的实现是可能的。

#### 多媒体****和DTV回放支持****

这部分介绍了Android的运行环境和应用框架的修改和扩展，以支持数字电视和多媒体设备控制。第一步将支持硬件音频和视频解码器，并允许本地多媒体文件回放（这一步是非常重要的，因为CPU发现DTV及多媒体解码器设备没有那么强大来做软件多媒体解码，而是使用集成多媒体硬件加速设备）。第二步将支持DTV（数字电视）频道搜索和记忆。第三步也是最后一步将增加对DTV（数字电视）频道回放和录制的支持。

1. **多媒体回放支持**

Android的Java运行库，拥有一个媒体对象“android.media”，是一个用于控制播放的音频/视频文件和流的接口。Android多媒体的实现在很大程度上依赖于一个开源软件“OpenCore”媒体管理库（最初Android使用的是PacketVideo），它执行多媒体数据所有的收集、加工、解码和编码任务。而这种做法深深的依赖于CPU的处理能力（这种处理能力是DTV和机顶盒平台很可能没有的）。

为了支持硬件加速多媒体回放，媒体框架将扩展选项，支持硬件处理、解码和呈现多媒体数据，但是只能通过事件通知Android（状态变化）。这将通过增加额外的硬件插件机制（进一步称为MMlib）来实现。



图2 MMLib库架构

MMlib库作为硬件加速平台解码器和Android多媒体对象之间的粘合层，位于“android\_media\_MediaPlayer”JNI对象中。MMLIB查询接口的初始化通过（为硬件支持信息）安装硬件平台插件以及存储相应的播放接口（插件）来实现。使用所提供的支持信息，当媒体播放对象发送多媒体指令时，MMlib解释处理此命令并选择最佳的播放接口来完成此次请求，如果没有硬件加速的支持，MMLIB直接将命令传到OPENCORE的库（保留在Android中定义的默认程序）。

Micronas平台支持多种视频和音频的格式，因此，基于这个项目的目的，对这些音频、视频和图像格式的支持将通过一些独立的插件来实现。

1. **DTV频道记忆及搜索**

DTV（数字电视）和机顶盒通常包括各种前端装置，来提供模拟和数字通道再现。为了使信道能够搜索和记忆，在Android的编程方式中，我们引入了两个Java类（一个利用对象的过程服务），一个用户接口和一个JNI接口对象，以支持本地信道服务。

引进的对象包括：

* **信道服务线程对象**

这个信道服务线程运行在后台进程中，使用了JNI接口。这项服务的主要功能是收集和修改有关支持平台的前端设备信息以及这些设备的能力。这项服务的辅助功能是异步提供对手动和自动搜台的支持，以及支持收集记忆频道中广播数据所提供的各种频道服务，另外还支持网络上提供这方面信息的插件。通过它所维护的ChannelServiceDatabase对象来存储和使用信息。

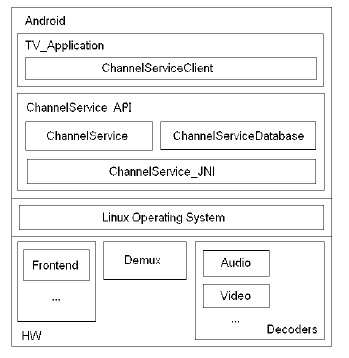


图3信道服务架构

* **ChannelServiceDatabase对象**

信道服务数据库提供了一个高度灵活的方式来存储关于所有不同信道类型的信息（DVBT，DVB-S，DVB-C，, Analogue， Radio等，以及他们所有的子类型和将来的类型）。结合使用SQL数据库，Android提供了一个很好的框架来灵活地存储数据。ChannelService -Database对象对最终用户并不是直接可见的，它的内容需要通过ClannelServiceClient接口来浏览和修改。

通道服务数据库的实现只提供了频道搜索和记忆的支持，但不是所有的Android设备都能够识别。识别的支持是通过通道数据库与本地的android文件系统同步，以及利用信道所提供的文件来实现的，这些文件有一个已知的MIME类型，这些文件解析为一个新的多媒体对象，且他们代表了Android多媒体框架的说明。

* **ChannelServiceClient接口**

通道服务客户端接口为通道服务子系统提供了一个入口，它实现了一些函数和对象，这些函数和对象是前端、通道和通道服务管理在控制用户使用它们的方式的时候常用的。因为Android同时可以有多个TV程序在运行，所以ChannelServiceClient接口必须实现异步方式来访问ChannelService子系统。

为了使用通道服务子系统，Android应用程序只需要实现ChannelServiceClient这个接口。

* **信道服务JNI接口**

该接口提供了本地平台的前端框架和Java信道服务之间的联系。它提供了查询平台的能力和获取前端、信道、信道服务信息的能力。这个接口编译连接成一个共享对象。它从Java程序中动态打开，以检查平台是否已支持信道服务。

1. **DTV频道的回放及录制**

通过扩大媒体对象框架支持MMlib库的能力，并通过定义通道服务的多媒体文件，DTV设备的控制和回放的实现，仅仅是一个MMlib的插件问题。

DTV回放和录制的MMlib插件的实现依赖于平台，在Micronas平台上DTV接口是直接通过内核驱动程序来实现的，所以实现DTV插件是很简单的，因为频道设置的指令，已经被频道服务的多媒体文件所接收。

#### Android TV 应用

Android平台没有DTV播放应用，然而，从之前的段落中可以看出，由于对Android软件栈的修改和补充，Android对DTV播放是支持的。

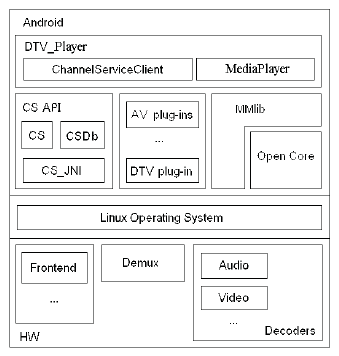


图4 DTV播放器应用程序概述

为了达到测试所有DTV平台能力的目的，实现了DTV播放器。

#### 七、测试结果

Android平台是专为移动设备而设计的，而且它还是一个正在广泛发展的软件平台。然而，当把它正确的移植在新系统上，它将自动成为能够执行的内置应用程序。尽管是为小屏幕的手持移动设备设计的，但是移植Android操作在测试型的DTV(数字电视)解码器设备上是可接受的。假设提供的测试平台有一个相当缓慢的CPU，GUI速度也是很少缓慢。这对于最新的机顶盒是没有问题的，因为它使用的Micronas平台有多达两倍的功率。

为了支持数字电视频道的搜索和回放，修改和扩展的Android软件栈已经用数字电视播放器应用程序进行了测试，测试包括频道的搜索、记忆、播放和录制。

记录在[6][7][8]的平台和目前的商业解决方案，多媒体只有有限数量的选项提供给用户，相对于他们来说，Android驱动平台能够让用户添加选项，修改的选项，换句话说， Android驱动平台将完全改变用户使用嵌入式多媒体设备的方式。

#### 八、结论

今天，Android并不打算只用在DTV平台上。然而，随着它的修改和扩展，它将为新的嵌入式多媒体设备提供非常好的软件环境。作为一个开源平台，Android可通过扩展、内核的修改、新应用的开发，为我们带来几乎无可限量的可能。本文为多媒体设备的开发提供了一种新的方式。在Micronas IDTV平台上，通过为Android使数字电视服务，再加上内置的Android应用程序（Web浏览器，JAVA游戏，图片，音频和视频管理等），电视机的使用方式将彻底改变。在嵌入式多媒体系统上使用Android软件栈（或一些未来的Andr​​oid衍生物）在不久的将来将成为现实，这将整体提升现代多媒体系统的体验。

#### 参考文献

1. 谷歌Android，<http://www.android.com>
2. Dalvik虚拟机，<http://www.dalvikvm.com>
3. 分析Dalvik虚拟机和类路径库，<http://imsciences.edu.pk/serg/wp-content/uploads/2009/07/Analysis-of-Dalvik-VM.pdf>
4. Micronas IDTV开发平台，

<http://www.micronas.com/pressroom/press_releases/articles/0812/index.html?newslang=1>

1. MIPS Android， <http://www.mips.com/android>
2. Lalit Kumar, Rajesh Kushwaha, Rishi Prakash，“设计与开发基于数字机顶盒浏览器的小型Linux操作系统”，2009年第一次关于计算智能，通信系统和网络的国际会议，277-281页
3. 李琼，郭民强，“基于嵌入式Linux操作系统的数字可记录综合电视“，2009年世界大会关于计算机科学与信息工程，81-84页
4. Ganesh Sivaraman, Pablo Cesar, Petri Vuorimaa “系统软件数字电视应用“，2001年IEEE国际会议多媒体与博览会（ICME'01），第154页
5. MIPS Android源代码，

<http://www.linuxfordevices.com/c/a/News/MIPS-to-release-Android-source-code-by-August/>